Amendments to the Claims:

This listing of the claims will replace all prior versions and listings of claims in the present application.

Listing of the Claims:

Claims 1-16. (Canceled).

17. (Currently Amended) A method for configuring a computer program including at least one functional unit, comprising:

at least one of:

creating at least one implementation-independent configuration data file, and

altering information filed in the at least one implementation-independent configuration data file;

wherein the information stored in the at least one implementation-independent configuration data file describes concrete configuration values in an abstract fashion;

using a computer script, at least one of automatically setting-up and automatically updating configuration data, stored in a configuration data container, as a function of the information filed in the at least one implementation-independent configuration data file, wherein the configuration data are extracted from the implementation-independent configuration data file and stored in the configuration data container;

automatically generating at least one item of dependency information describing a dependency on at least two configuration data present in the configuration data container ~~at least one of:~~

~~whether a particular resource is reserved exclusively for use by the least one functional unit, which comprises a software module in the computer program; and~~

~~a sequence in which additional computer scripts, which alter the configuration data stored in the configuration data container, must be executed~~;

automatically generating at least one implementation-dependent configuration data file as a function of the configuration data stored in the configuration data container, and as a function of the at least one item of dependency information, wherein concrete configuration data are stored in the implementation-dependent configuration data file; and automatically configuring the at least one functional unit as a function of concrete information filed in the at least one implementation-dependent configuration data file, wherein each of the automated steps above are performed at a processor of a computer.

18. (Canceled).

19. (Previously Presented) The method as recited in Claim 17, further comprising:

creating a plurality of implementation-independent configuration data files; and

assigning each of the implementation-independent configuration data files to the at least one functional unit.

20. (Previously Presented) The method as recited in Claim 17, further comprising:

generating a plurality of implementation-dependent configuration data files, and

assigning each of the implementation-dependent configuration data files to the at least one functional unit.

21. (Previously Presented) The method as recited in Claim 20, wherein the at least one implementation-dependent configuration data file is generated as a function of at least one property of hardware on which an installation of at least a portion of the configured computer program is to be made possible.

22. (Previously Presented) The method as recited in Claim 20, wherein the at least one implementation-dependent configuration data file is generated as a function of a result of a plausibility check in which it is determined:

whether the hardware is capable of providing data required by the at least one functional unit; and

whether a resource required by the at least one functional unit is available.

23. (Canceled)

24. (Previously Presented) The method as recited in Claim 20, further comprising:

automatically creating a documentation that describes the information filed within at least one of the at least one implementation-independent configuration data file and the at least one implementation-dependent configuration data file.

25. (Previously Presented) The method as recited in Claim 17, wherein the at least one implementation-independent configuration data file is created in an XML-based format.

26. (Previously Presented) The method as recited in Claim 17, further comprising:

automatically determining, as a function of the configuration data, whether a functional unit included by the computer program is needed by the computer program, wherein the functional unit is only configured if the functional unit is needed by the computer program.

27. (Currently Amended) A non-transitory computer readable storage medium storing a software system for configuring a computer program including at least one functional unit, the software system comprising:

at least one implementation-independent configuration data file, wherein information stored in the at least one implementation-independent configuration data file describes concrete configuration values in an abstract fashion;

at least one of:

a configuration data container including configuration data, and

an arrangement for creating the configuration data container as a function of information filed in the at least one implementation-independent configuration data file;

wherein the configuration data are extracted from the implementation-independent configuration data file and stored in the configuration data container;

an arrangement that includes a computer script for at least one of altering and reading out configuration data from the configuration data container;

an arrangement for automatically generating at least one item of dependency information describing a dependency on at least two configuration data present in the configuration data container ~~at least one of:~~

~~whether a particular resource is reserved exclusively for use by the least one functional unit, which comprises a software module in the computer program; and~~

~~a sequence in which additional computer scripts, which alter the configuration data stored in the configuration data container, must be executed~~;

an arrangement for automatically generating at least one implementation-dependent configuration data file as a function of configuration data stored in the configuration data container, and as a function of the at least one item of dependency information<u>, wherein concrete configuration data are stored in the implementation-dependent configuration data file</u>; and

an arrangement for automatically configuring the at least one functional unit as a function of <u>concrete</u> information filed in the implementation-dependent configuration data file.

28. (Previously Presented) The storage medium as recited in Claim 27, the software system further comprising:

an arrangement for at least one of:

creating the at least one implementation-independent configuration data file, and

altering information filed in the at least one implementation-independent configuration data file;

an arrangement for at least one of automatically setting-up and automatically updating configuration data, stored in the configuration data container, as a function of the information filed in the at least one implementation-independent configuration data file;

an arrangement for automatically generating at least one implementation-dependent configuration data file as a function of the configuration data stored in the configuration data container; and

an arrangement for automatically configuring the at least one functional unit as a function of information filed in the at least one implementation-dependent configuration data file.

29. (Canceled).

30. (Previously Presented) The storage medium as recited in Claim 27, wherein the storage medium is one of a random access memory, a read-only memory, and a flash memory.

31. (Previously Presented) The storage medium as recited in Claim 27, wherein the storage medium is one of a digital versatile disk, a compact disk, and a hard disk.

32. (Currently Amended) A computing element having a microprocessor and being programmed with software that when executed results in a performance of the following:

at least one of:

creating at least one implementation-independent configuration data file, and

altering information filed in the at least one implementation-independent configuration data file;

wherein information stored in the at least one implementation-independent configuration data file describes concrete configuration values in an abstract fashion;

using a computer script, at least one of automatically setting-up and automatically updating configuration data, stored in a configuration data container, as a function of the information filed in the at least one implementation-independent configuration data file, wherein the configuration data are extracted from the implementation-independent configuration data file and stored in the configuration data container;

automatically generating at least one item of dependency information describing a dependency on at least two configuration data present in the configuration data container ~~at least one of:~~

~~whether a particular resource is reserved exclusively for use by the least one functional unit, which comprises a software module in the computer program; and~~

~~a sequence in which additional computer scripts, which alter the configuration data stored in the configuration data container, must be executed~~;

automatically generating at least one implementation-dependent configuration data file as a function of the configuration data stored in the configuration data container, and as a function of the at least one item of dependency information, wherein concrete configuration data are stored in the implementation-dependent configuration data file; and automatically configuring the at least one functional unit as a function of <u>concrete</u> information filed in the at least one implementation-dependent configuration data file.

33. (Previously Presented) The computing element as recited in Claim 32, wherein the computing element corresponds to a control device.

34. (New) A method for configuring a computer program including at least one functional unit, comprising:

at least one of:

creating at least one implementation-independent configuration data file, and

altering information filed in the at least one implementation-independent configuration data file;

using a computer script, at least one of automatically setting-up and automatically updating configuration data, stored in a configuration data container, as a function of the information filed in the at least one implementation-independent configuration data file;

automatically generating at least one item of dependency information describing a sequence in which additional computer scripts, which alter the configuration data stored in the configuration data container, must be executed;

automatically generating at least one implementation-dependent configuration data file as a function of the configuration data stored in the configuration data container, and as a function of the at least one item of dependency information; and

automatically configuring the at least one functional unit as a function of information filed in the at least one implementation-dependent configuration data file, wherein each of the automated steps above are performed at a processor of a computer.

35. (New) A method for configuring a computer program including at least one functional unit, comprising:

at least one of:

creating at least one implementation-independent configuration data file, and

altering information filed in the at least one implementation-independent configuration data file;

using a computer script, at least one of automatically setting-up and automatically updating configuration data, stored in a configuration data container, as a function of the information filed in the at least one implementation-independent configuration data file;

automatically generating at least one item of dependency information describing whether a particular resource is reserved exclusively for use by the at least one functional unit, which comprises a software module in the computer program;

automatically generating at least one implementation-dependent configuration data file as a function of the configuration data stored in the configuration data container, and as a function of the at least one item of dependency information; and

automatically configuring the at least one functional unit as a function of information filed in the at least one implementation-dependent configuration data file, wherein each of the automated steps above are performed at a processor of a computer.